

**METHOD AND APPARATUS FOR PIPELINED JOINT EQUALIZATION AND  
DECODING FOR GIGABIT COMMUNICATIONS**

**Cross Reference to Related Applications**

5           This application claims the benefit of United States Provisional Application  
Number 60/245,519, filed November 3, 2000.

**Field of the Invention**

10           The present invention relates generally to channel equalization and decoding  
techniques, and more particularly, to sequence estimation techniques with shorter critical paths.

**Background of the Invention**

15           The transmission rates for local area networks (LANs) that use unshielded twisted  
pair (UTP) copper cabling have progressively increased from 10 Megabits-per-second (Mbps) to  
1 Gigabit-per-second (Gbps). The Gigabit Ethernet 1000 Base-T standard, for example, operates  
at a clock rate of 125 MHz and uses UTP cabling of Category 5 with four pairs to transmit 1  
Gbps. Trellis-coded modulation (TCM) is employed by the transmitter, in a known manner, to  
achieve coding gain. The signals arriving at the receiver are typically corrupted by intersymbol  
interference (ISI), crosstalk, echo, and noise. A major challenge for 1000 Base-T receivers is to  
20           jointly equalize the channel and decode the corrupted trellis-coded signals at the demanded clock  
rate of 125 MHz, as the algorithms for joint equalization and decoding incorporate non-linear  
feedback loops that cannot be pipelined.

25           Data detection is often performed using maximum likelihood sequence  
estimation, to produce the output symbols or bits. A maximum likelihood sequence estimator  
considers all possible sequences and determines which sequence was actually transmitted, in a  
known manner. The maximum likelihood sequence estimator is the optimum decoder and  
applies the well-known Viterbi algorithm to perform joint equalization and decoding. For a more  
detailed discussion of a Viterbi implementation of a maximum likelihood sequence estimator  
(MLSE), see Gerhard Fettweis and Heinrich Meyr, "High-Speed Parallel Viterbi Decoding  
30           Algorithm and VLSI-Architecture," IEEE Communication Magazine (May 1991), incorporated  
by reference herein.

In order to reduce the hardware complexity for the maximum likelihood sequence estimator that applies the Viterbi algorithm, a number of sub-optimal approaches which are referred to as reduced-state sequence estimation (RSSE) have been proposed. For a discussion of reduced state sequence estimation techniques, as well as the special cases of decision-feedback sequence estimation (DFSE) and parallel decision-feedback decoding (PDFD) techniques, see, 5 for example, P. R. Chevillat and E. Eleftheriou, "Decoding of Trellis-Encoded Signals in the Presence of Intersymbol Interference and Noise", IEEE Trans. Commun., vol. 37, 669-76, (July 1989), M. V. Eyuboglu and S. U. H. Qureshi, "Reduced-State Sequence Estimation For Coded Modulation On Intersymbol Interference Channels", IEEE JSAC, vol. 7, 989-95 (Aug. 1989), or 10 A. Duel-Hallen and C. Heegard, "Delayed Decision-Feedback Sequence Estimation," IEEE Trans. Commun., vol. 37, pp. 428-436, May 1989, each incorporated by reference herein.

Generally, reduced state sequence estimation techniques reduce the complexity of the maximum likelihood sequence estimators by merging several states. The RSSE technique incorporates non-linear feedback loops that cannot be pipelined. The critical path associated 15 with these feedback loops is the limiting factor for high-speed implementations.

United States Patent Application Serial Number 09/326,785, filed June 4, 1999 and entitled "Method and Apparatus for Reducing the Computational Complexity and Relaxing the Critical Path of Reduced State Sequence Estimation (RSSE) Techniques," incorporated by reference herein, discloses a technique that reduces the hardware complexity of RSSE for a given 20 number of states and also relaxes the critical path problem. United States Patent Application Serial Number 09/471,920, filed December 23, 1999, entitled "Method and Apparatus for Shortening the Critical Path of Reduced Complexity Sequence Estimation Techniques," incorporated by reference herein, discloses a technique that improves the throughput of RSSE by pre-computing the possible values for the branch metrics in a look-ahead fashion to permit 25 pipelining and the shortening of the critical path. The complexity of the pre-computation technique, however, increases exponentially with the length of the channel impulse response. In addition, the delay through the selection circuitry that selects the actual branch metrics among all precomputed ones increases with  $L$ , eventually neutralizing the speed gain achieved by the precomputation.

A need therefore exists for a technique that increases the throughput of RSSE algorithms using precomputations with only a linear increase in hardware complexity with respect to the look-ahead computation depth.

## 5 Summary of the Invention

Generally, a method and apparatus are disclosed for the implementation of reduced state sequence estimation with an increased throughput using precomputations (look-ahead), while only introducing a linear increase in hardware complexity with respect to the look-ahead depth. RSSE techniques typically decode a received signal and compensate for intersymbol interference using a decision feedback unit (DFU), a branch metrics unit (BMU), an add-compare-select unit (ACSU) and a survivor memory unit (SMU). The present invention limits the increase in hardware complexity by taking advantage of past decisions. The past decision may be a past ACS decision of the ACSU or a past survivor symbol in the SMU or a combination thereof. The critical path of a conventional RSSE implementation is broken up into at least two smaller critical paths using pipeline registers.

A reduced state sequence estimator is disclosed that employs a one-step look-ahead technique to process a signal received from a dispersive channel having a channel memory. Initially, a speculative intersymbol interference estimate is precomputed based on a combination of (i) a speculative partial intersymbol interference estimate for a first postcursor tap of the channel impulse response, based on each possible value for a data symbol, and (ii) a combination of partial intersymbol interference estimates for each subsequent postcursor tap of the channel impulse response, where at least one of the partial intersymbol interference estimates for the subsequent postcursor taps is based on a past survivor symbol from the corresponding state. In addition, a branch metric is precomputed based on the precomputed intersymbol interference estimate. One of the precomputed branch metrics is selected based on a past decision from the corresponding state. The past decision may be a past ACS decision of the ACSU or a past survivor symbol in the SMU or a combination of both. The selected branch metric is used to compute new path metrics for path extensions from a corresponding state. The computed new path metrics are used to determine the best survivor path and path metric for a corresponding state.

A reduced state sequence estimator is also disclosed that employs a multiple-step look-ahead technique to process a signal received from a dispersive channel having a channel memory. Initially, a speculative partial intersymbol interference estimate is precomputed for each of a plurality of postcursor taps of the channel impulse response, based on each possible value for a data symbol. Thereafter, a partial intersymbol interference estimate is selected for each of the plurality of postcursor taps other than a first postcursor tap based on a past decision from a corresponding state. The past decision may be a past ACS decision of the ACSU or a past survivor symbol in the SMU or a combination of both. A precomputed partial intersymbol interference estimate for the first postcursor tap is referred to as a precomputed intersymbol interference estimate. In addition, speculative branch metrics are precomputed based on the precomputed intersymbol interference estimates. One of the precomputed branch metrics is selected based on a past decision from a corresponding state. The past decision may be a past ACS decision of the ACSU or a past survivor symbol in the SMU or a combination of both. The selected branch metric is used to compute new path metrics for path extensions from a corresponding state. The computed new path metrics are used to determine the best survivor path and path metric for a corresponding state.

In further variations, intersymbol estimates can be selected among precomputed intersymbol interference estimates without precomputing branch metrics or the partial intersymbol interference estimates can be precomputed for a group of taps, with a precomputation for all possible data symbol combinations corresponding to the groups of taps and selection for each group.

A more complete understanding of the present invention, as well as further features and advantages of the present invention, will be obtained by reference to the following detailed description and drawings.

### **Brief Description of the Drawings**

FIG. 1 illustrates a channel impulse response with channel memory,  $L$ ;

FIG. 2 illustrates a communication system in which the present invention may operate;

FIG. 3 illustrates a trellis associated with a channel of memory length  $L=1$  and binary data symbols;

FIG. 4 illustrates a block diagram for an implementation of the Viterbi algorithm (VA);

FIG. 5 illustrates a state-parallel implementation of the ACSU of FIG. 4 for a channel of memory  $L=1$  ;

FIG. 6 is a table analyzing the complexity and critical path of MLSE and RSSE techniques;

FIG. 7A illustrates the architecture of a reduced state sequence estimator;

FIG. 7B illustrates an implementation of the look-up tables in the BMU of FIG. 7A;

FIG. 8 illustrates an exemplary look-ahead architecture for an RSSE algorithm with one-step look-ahead in accordance with one embodiment of the present invention;

FIG. 9 illustrates an exemplary look-ahead architecture for an RSSE algorithm with multiple-step look-ahead in accordance with another embodiment of the present invention;

FIG. 10 is a table analyzing the complexity and critical path of a pipelined RSSE in accordance with the present invention;

FIGS. 11 and 12 illustrate alternate implementations of the RSSE algorithms with one-step look-ahead (FIG. 8) and multiple-step look-ahead (FIG. 9), respectively;

FIG. 13 illustrates a trellis for a multi-dimensional trellis code, such as the 1000BASE-T trellis code;

FIG. 14 is a schematic block diagram illustrating a pipelined parallel decision feedback decoder (PDFD) architecture that decodes the 1000BASE-T trellis code and equalizes intersymbol interference in accordance with the present invention;

FIG. 15 is a schematic block diagram illustrating an embodiment of the look-ahead decision feedback unit (LA-DFU) of FIG. 14;

FIG. 16 is a schematic block diagram illustrating an embodiment of the intersymbol interference selection unit (ISI-MUXU) of FIG. 14;

FIG. 17 is a schematic block diagram illustrating an embodiment of the one dimensional look-ahead branch metrics unit (1D-LA-BMU) of FIG. 14; and

FIG. 18 is a schematic block diagram illustrating an embodiment of the survivor memory unit (SMU) of FIG. 14.

### Detailed Description

As previously indicated, the processing speed of conventional reduced state sequence estimation (RSSE) implementations is limited by a recursive feedback loop. According to one feature of the present invention, the processing speed of reduced state sequence estimation implementations is improved by pipelining the branch metric and decision-feedback computations, such that the critical path is reduced to be of the same order as in a traditional Viterbi decoder. The additional hardware required by the present invention scales only linearly with the look-ahead depth. The presented algorithm allows the VLSI implementation of RSSE for high-speed applications such as Gigabit Ethernet over copper. Reduced complexity sequence estimation techniques are disclosed for uncoded signals, where the underlying trellis has no parallel state transitions, as well as for signals encoded with a multi-dimensional trellis code having parallel transitions, such as signals encoded according to the 1000BASE-T Ethernet standard. It should be understood that the disclosed pipelining technique can be applied whenever RSSE is being used, e.g., to any kind of trellis or modulations scheme. The disclosed examples are used for illustration purposes only and do not intend to limit the scope of the invention.

### System Model

FIG. 1 illustrates a channel impulse response with channel memory,  $L$ . As shown in FIG. 1, there is a main tap corresponding to time 0, and there are  $L$  postcursor taps. The first  $K$  postcursor taps shown in FIG. 1 after the main tap are used for the construction of the reduced-state trellis, as discussed below.

FIG. 2 illustrates a communication system 200 having a channel 210 and a sequence estimator 220. The output of the channel 210 at time  $n$  is given by

$$z_n = \sum_{i=0}^L f_i \cdot a_{n-i} + w_n, \quad (1)$$

where  $\{f_i\}$ ,  $0 \leq i \leq L$  are the finite impulse response channel coefficients ( $f_0 = 1$  is assumed without loss of generality),  $L$  is the channel memory,  $a_n$  is the data symbol at time  $n$ , and  $w_n$  is

zero-mean Gaussian noise. The decision of the sequence estimator 220 corresponding to  $a_n$  is denoted by  $a'_n$ . While the illustrative embodiment assumes that the symbols are binary, i.e.,  $a_n = \{-1, 1\}$ , and trellis-coded modulation (TCM) is not employed. The present invention may be applied, however, to non-binary modulation and TCM, such as the coding and modulation scheme used in Gigabit Ethernet over copper, as would be apparent to a person of ordinary skill in the art.

The optimum method for the recovery of the transmitted symbols is MLSE, which applies the Viterbi algorithm (VA) to the trellis defined by the channel state

$$\rho_n = (a_{n-1}, a_{n-2}, \dots, a_{n-L}). \quad (2)$$

A binary symbol constellation is assumed. Thus, the number of states processed by the VA is given by:

$$S = 2^L, \quad (3)$$

and two branches leave or enter each state of the trellis. FIG. 3 shows a trellis associated with a channel of memory length  $L=1$ . The branch metric for a transition from state  $\rho_n$  under input  $a_n$  is given by

$$\lambda_n(z_n, a_n, \rho_n) = (z_n - a_n - \sum_{i=1}^L f_i a_{n-i})^2. \quad (4)$$

The VA determines the best survivor path into state  $\rho_{n+1}$  from the two predecessor states  $\{\rho_n\}$  by evaluating the following add-compare-select (ACS) function:

$$\Gamma_{n+1}(\rho_{n+1}) = \min_{\{\rho_n\} \rightarrow \rho_{n+1}} (\Gamma_n(\rho_n) + \lambda_n(z_n, a_n, \rho_n)), \quad (5)$$

where  $\Gamma_n(\rho_n)$  is the path metric for state  $\rho_n$ .

The block diagram for an implementation of the VA is shown in FIG. 4. As shown in FIG. 4, the VA 400 includes a branch metrics unit (BMU) 410, an add-compare-select unit (ACSU) 420 and a survivor memory unit (SMU) 430. The BMU 410 calculates the  $2^{L+1}$  branch metrics (BMs), the ACSU 420 performs the ACS operation for each of the  $S$  states, and the SMU 430 keeps track of the  $S$  survivor paths.

The ACSU 420 is the bottleneck for maximum throughput as the operations in the BMU 410 and SMU 430 are feedforward and can thus be pipelined using pipeline registers 415 and 425. A state-parallel implementation of the ACSU 420 yields the highest processing speed

and is shown in FIG. 5 for a channel of memory  $L=1$  (the corresponding trellis was shown in FIG. 3).

The recursive loop of the ACS operation associated with equation (5) determines the critical path in the ACSU 420, as it cannot be pipelined. It can be seen from FIG. 5 that this loop comprises one addition (ADD) 510, one 2-way comparison 520, whose delay is about the same as one ADD, and a 2-way selection 530, corresponding to a 2-to-1 multiplexer (MUX). Hereinafter, shift registers will not be considered in the critical path analysis due to their minor delay. FIG. 6 is a table 600 analyzing the complexity and critical path of MLSE and RSSE. Column 620 of table 600 summarizes the computational complexity and critical path of MLSE for binary signals corrupted by a channel of memory  $L$ . It is noted that in addition to a state-parallel implementation shown in FIG. 5, the throughput of the VA can be even further increased by introducing parallelism on the bit, block and algorithmic level (for a good summary, see e.g. H. Meyr, M. Moeneclaey, and S.A. Fechtel, Digital Communication Receivers, John Wiley & Sons, pp. 568-569, 1998). However, this comes at a significant increase in complexity and/or latency.

#### Reduced-State Sequence Estimation

RSSE reduces the complexity of MLSE by truncating the channel memory  $\rho_n$ , as described in A. Duel-Hallen and C. Heegard, "Delayed Decision-Feedback Sequence Estimation," IEEE Trans. Commun., vol. 37, 428-436, May 1989, or applying set partitioning to the signal alphabet as described in P. R. Chevillat and E. Eleftheriou, "Decoding of Trellis-Encoded Signals in the Presence of Intersymbol Interference and Noise," IEEE Trans. Commun., vol. 37, pp. 669-676, Jul. 1989 or M.V. Eyuboglu and S.U. Qureshi, "Reduced-State Sequence Estimation for Coded Modulation on Intersymbol Interference Channels," IEEE J. Sel. Areas Commun., vol. 7, pp. 989-995, Aug. 1989. Similar to the VA, RSSE searches for the most likely data sequence in the reduced trellis by keeping only the best survivor path for each reduced state. In the exemplary embodiment discussed herein, the reduced state  $\rho'_n$  is obtained by truncating equation (2) to  $K$  yielding

$$\rho'_n = (a_{n-1}, a_{n-2}, \dots, a_{n-K}), \quad 0 \leq K \leq L. \quad (6)$$

In this case, the number of reduced states is given by

$$S' = 2^K. \quad (7)$$



The results may be generalized to the cases given in P. R. Chevillat and E. Eleftheriou or M.V. Eyuboglu and S.U. Qureshi, referenced above. The branch metric for a transition from reduced state  $\rho'_n$  under input  $a_n$  is given by

$$\lambda'_n(z_n, a_n, \rho'_n) = (z_n - a_n + u_n(\rho'_n))^2, \quad (8)$$

5 where

$$u_n(\rho'_n) = -\sum_{i=1}^K f_i a_{n-i} - \sum_{i=K+1}^L f_i \hat{a}_{n-i}(\rho'_n). \quad (9)$$

In equation (9),  $u_n(\rho'_n)$  is the decision-feedback for  $\rho'_n$  and  $\hat{a}_{n-i}(\rho'_n)$  is the symbol of the survivor path into state  $\rho'_n$  which corresponds to time  $n-i$ . As the first  $K$  survivor symbols  $(\hat{a}_{n-1}(\rho'_n), \hat{a}_{n-2}(\rho'_n), \dots, \hat{a}_{n-K}(\rho'_n))$  from the survivor path into state  $\rho'_n$  are equal to the symbols  $(a_{n-1}, a_{n-2}, \dots, a_{n-K})$  defining this state, equation (9) can be rewritten as

$$u_n(\rho'_n) = -\sum_{i=1}^L f_i \hat{a}_{n-i}(\rho'_n). \quad (10)$$

Among all paths entering reduced state  $\rho'_{n+1}$  from the 2 predecessor states  $\{\rho'_n\}$ , the most likely path with metric  $\Gamma'_{n+1}(\rho'_{n+1})$  is chosen according to the ACS operation:

$$\Gamma'_{n+1}(\rho'_{n+1}) = \min_{\{\rho'_n\} \rightarrow \rho'_{n+1}} (\Gamma'_n(\rho'_n) + \lambda'_n(z_n, a_n, \rho'_n)). \quad (11)$$

15 The state-parallel architecture for RSSE with the parameters  $L=4$  and  $K=1$  is shown in FIG. 7A. It can be seen from FIG. 7A that the RSSE 700 architecture comprises four functional blocks, namely, a decision feedback unit (DFU) 710, a branch metrics unit (BMU) 720, an add-compare-select unit (ACSU) 730 and a survivor memory unit (SMU) 740. As the corresponding reduced trellis is the same as the one in FIG. 3, the ACSU 730 shown in FIG. 7 has the same architecture as the ACSU 420 given in FIG. 5. The part of the SMU 740 that stores the  $L-K$  survivor symbols  $(\hat{a}_{n-K-1}(\rho'_n), \hat{a}_{n-K-2}(\rho'_n), \dots, \hat{a}_{n-L}(\rho'_n))$  for each reduced state must be implemented in a register-exchange-architecture as described in R. Cypher and C.B. Shung, "Generalized trace-back techniques for survivor memory management in the Viterbi algorithm," J. VLSI Signal Processing, vol. 5, pp. 85-94, 1993, as these symbols are required for the evaluation of equation (9) in the DFU 710 without delay. Because of the binary modulation, the multipliers in the DFU 710 can be implemented using shifters (SHIFTS). Look-up tables (LUTs) approximate the squaring function in equation (8) in the BMU, as defined by FIG. 7B.

RSSE 700 has less computational complexity than MLSE for the same channel memory  $L$ , as RSSE processes less states, at the expense of a significantly longer critical path. It can be seen from FIG. 7 that there is a recursive loop which comprises one SHIFT and  $L-K+1$  ADDs in the DFU 710 (the first term in the right hand side of equation (9) can be computed outside the loop), one LUT in the BMU 720, one add-compare in the ACSU 730 (which is roughly equal to two ADDs in terms of delay), and a 2-to-1 MUX in the SMU 740. All these operations must be completed within one symbol period and cannot be pipelined. In contrast to this, the critical path in MLSE just comprises the ACS operation. Also, due to the different structure of the recursive loop in the RSSE 700, the block processing methods which have been developed to speed up the VA (see H. Meyr et al., Digital Communication Receivers, John Wiley & Sons, 568-569 (1998)) cannot be applied to increase the throughput of RSSE. Therefore, the maximum throughput of RSSE is potentially significantly lower than of MLSE. Furthermore, the throughput of RSSE depends on the channel memory such that it decreases for increasing  $L$ . FIG. 6 summarizes the comparison of MLSE and RSSE in terms of computational complexity and critical path.

#### Pipelined RSSE

It was suggested in E.F. Haratsch and K. Azadet, "High-speed reduced-state sequence estimation," Proc. IEEE Int. Symp. Circuits and Systems, May 2000, to precompute the branch metrics for all possible  $2^L$  channel states  $\rho_n$  in a look-ahead fashion outside the critical loop. At each decoding step, the appropriate branch metrics are chosen based on past survivor symbols in the SMU. This approach removes the BMU and DFU out of the critical loop. However, the hardware increases exponentially with the channel memory  $L$ . Also the delay through the MUXs, which select the actual branch metrics among all precomputed ones, increases with  $L$ , eventually neutralizing the speed gain achieved by the precomputation. The present invention provides a technique that increases the throughput of RSSE by performing precomputations while only leading to a linear increase in hardware with respect to the look-ahead depth.

#### One-Step Look-Ahead

The hardware increase can be limited by taking advantage of past survivor symbols in the SMU and past decisions of the ACSU. This will be shown for precomputations

with look-ahead depth one, i.e. possible values for branch metrics needed by the ACSU at time  $n$  are already computed at time  $n-1$ .

A partial decision-feedback for reduced state  $\rho'_n$  could be calculated by using the  $L-1$  survivor symbols  $(\hat{a}_{n-2}(\rho'_{n-1}), \hat{a}_{n-3}(\rho'_{n-1}), \dots, \hat{a}_{n-L}(\rho'_{n-1}))$  corresponding to the survivor sequence into  $\rho'_{n-1}$ :

$$v_n(\rho'_{n-1}) = -\sum_{i=2}^L f_i \hat{a}_{n-i}(\rho'_{n-1}). \quad (12)$$

Note, that the  $K$  survivor symbols  $(\hat{a}_{n-2}(\rho'_{n-1}), \hat{a}_{n-3}(\rho'_{n-1}), \dots, \hat{a}_{n-K-1}(\rho'_{n-1}))$  need not to be fed back from the SMU, as they are equal to the symbols defining the state  $\rho'_{n-1}$  (c.f. equation (6)). Therefore, these symbols and their contribution to the partial decision-feedback  $v_n(\rho'_{n-1})$  are fixed for a particular state  $\rho'_{n-1}$ .

If  $\tilde{a}_{n-1}$  denotes a possible extension of the sequence  $(\hat{a}_{n-2}(\rho'_{n-1}), \hat{a}_{n-3}(\rho'_{n-1}), \dots, \hat{a}_{n-L}(\rho'_{n-1}))$ , the corresponding tentative decision-feedback is given by

$$\tilde{u}_n(\rho'_{n-1}, \tilde{a}_{n-1}) = v_n(\rho'_{n-1}) - f_1 \tilde{a}_{n-1}, \quad (13)$$

and the tentative branch metric under input  $a_n$  is

$$\tilde{\lambda}'_n(z_n, a_n, \rho'_{n-1}, \tilde{a}_{n-1}) = (z_n - a_n + \tilde{u}_n(\rho'_{n-1}, \tilde{a}_{n-1}))^2. \quad (14)$$

The actual branch metric corresponding to survivor paths into state  $\rho'_n$  and input  $a_n$  can be selected among the tentative branch metrics based on the past decision  $d_{n-1}(\rho'_{n-1} = \rho'_n)$  according to

$$\lambda'_n(z_n, a_n, \rho'_n) = \text{sel}(\Lambda_n(z_n, a_n, \rho'_n), d_{n-1}(\rho'_{n-1} = \rho'_n)), \quad (15)$$

where  $\Lambda_n(z_n, a_n, \rho'_n)$  is the vector containing the two tentative branch metrics  $\tilde{\lambda}'_n(z_n, a_n, \rho'_{n-1}, \tilde{a}_{n-1})$  for input  $a_n$  and the two possible sequences into  $\rho'_n$  from the different predecessor states  $\{\rho'_{n-1}\}$ :

$$\Lambda_n(z_n, a_n, \rho'_n) = \{\tilde{\lambda}'_n(z_n, a_n, \rho'_{n-1}, \tilde{a}_{n-1})\}, \{\rho'_{n-1}\} \rightarrow \rho'_n. \quad (16)$$

The branch metrics, which have been selected using equation (15), are used for the ACS operation according to equation (11). As equations (12), (13), (14), (15) and (16) can already be evaluated at time  $n-1$ , they are decoupled from the ACS operation according to equation (11) at time  $n$ . This leads to an architecture that can achieve a potentially higher throughput than the conventional RSSE implementation. The look-ahead architecture for the RSSE 700 of FIG. 7 (i.e.  $L=4$  and  $K=1$ ) is shown in FIG. 8. It can be seen that the long critical path in the architecture

of FIG. 7 is broken up into two smaller critical paths, as a pipeline stage 825 is placed in front of the ACSU 830. The processing speed of this architecture still depends on the channel memory, as the number of additions and thus the delay along the critical path in the DFU 810 increases with  $L$ . In the following, a pipelined RSSE architecture is discussed whose maximum throughput does not depend on  $L$ .

#### Multiple-Step Look-Ahead

The process of precomputing branch metrics which are needed at time  $n$  could already be started at time  $n-M$ , where  $M \in [1; L-K]$ . A partial decision-feedback corresponding to the survivor sequence  $(\hat{a}_{n-M-1}(\rho'_{n-M}), \hat{a}_{n-M-2}(\rho'_{n-M}), \dots, \hat{a}_{n-L}(\rho'_{n-M}))$  into  $\rho'_{n-M}$  is given by

$$v_n(\rho'_{n-M}) = -\sum_{i=M+1}^L f_i \hat{a}_{n-i}(\rho'_{n-M}). \quad (17)$$

It is again noted that the  $K$  survivor symbols  $(\hat{a}_{n-M-1}(\rho'_{n-M}), \hat{a}_{n-M-2}(\rho'_{n-M}), \dots, \hat{a}_{n-M-K}(\rho'_{n-M}))$  are identical to the symbols defining the state  $\rho'_{n-M}$ , and thus their contribution to  $v_n(\rho'_{n-M})$  is fixed for this particular state. A tentative partial decision-feedback for a sequence starting with  $(\hat{a}_{n-M-1}(\rho'_{n-M}), \hat{a}_{n-M-2}(\rho'_{n-M}), \dots, \hat{a}_{n-L}(\rho'_{n-M}))$  and which is extended by  $\tilde{a}_{n-M}$  can be precomputed as

$$\tilde{u}_n(\rho'_{n-M}, \tilde{a}_{n-M}) = v_n(\rho'_{n-M}) - f_M \tilde{a}_{n-M}. \quad (18)$$

When the decision  $d_{n-M}(\rho'_{n-M} = \rho'_{n-M+1})$  becomes available, the partial decision-feedback, which corresponds to the survivor sequence  $(\hat{a}_{n-M}(\rho'_{n-M+1}), \hat{a}_{n-M-1}(\rho'_{n-M+1}), \dots, \hat{a}_{n-L}(\rho'_{n-M+1}))$ , can be selected among the precomputed ones:

$$v_n(\rho'_{n-M+1}) = \text{sel}(U_n(\rho'_{n-M+1}), d_{n-M}(\rho'_{n-M} = \rho'_{n-M+1})), \quad (19)$$

where  $U_n(\rho'_{n-M+1})$  is the vector containing the two precomputed tentative partial decision-feedback values for the two possible path extensions into  $\rho'_{n-M+1}$  from the different predecessor states  $\{\rho'_{n-M}\}$ :

$$U_n(\rho'_{n-M+1}) = \{\tilde{u}_n(\rho'_{n-M}, \tilde{a}_{n-M})\}, \{\rho'_{n-M}\} \rightarrow \rho'_{n-M+1}. \quad (20)$$

To be able to eventually precompute tentative branch metrics according to equation (14), the computations described by equations (18), (19) and (20) must be repeated for time steps  $n-M+1$  to  $n-1$  according to the following equations, where  $M-1 \geq k \geq 1$ :

$$\tilde{u}_n(\rho'_{n-k}, \tilde{a}_{n-k}) = v_n(\rho'_{n-k}) - f_k \tilde{a}_{n-k}, \quad (21)$$

$$v_n(\rho'_{n-k+1}) = \text{sel}(U_n(\rho'_{n-k+1}), d_{n-k}(\rho'_{n-k} = \rho'_{n-k+1})), \quad (22)$$

$$U_n(\rho'_{n-k+1}) = \{\tilde{u}_n(\rho'_{n-k}, \tilde{a}_{n-k})\}, \{\rho'_{n-k}\} \rightarrow \rho_{n-k+1}. \quad (23)$$

Once  $\tilde{u}_n(\rho'_{n-1}, \tilde{a}_{n-1})$  becomes available, tentative branch metrics  $\tilde{\lambda}'_n(z_n, a_n, \rho'_{n-1}, \tilde{a}_{n-1})$  can be precomputed according to equation (14) and the appropriate branch metrics are selected according to equations (15) and (16).

The architecture for RSSE 900 with look-ahead depth  $M=3$  and the parameters  $L=4$  and  $K=1$  is shown in FIG. 9. It can be seen that in total  $M=3$  pipeline stages are available. Two pipeline stages 912, 916 have been placed inside the DFU 910, and one pipeline stage 925 has been placed between the BMU 920 and ACSU 930. The connection network in the DFU 910 resembles the structure of the underlying trellis from FIG. 3, as past decisions from the ACSU 930 are used to extend the partial survivor sequences by the subsequent survivor symbol. As the LUT typically has a delay comparable to an adder, the critical path of this implementation is determined by an add-compare in the ACSU 930 (2 ADDs) and the storage of the most recent decision in the SMU 940 or the selection of an appropriate value with a 2-to-1 MUX in the DFU 910 or BMU 920.

The complexity and critical path of pipelined RSSE using multiple-step look-ahead computations is shown in FIG. 10. It can be seen in FIG. 10 that the hardware overhead for performing precomputations scales only linearly with the look-ahead depth  $M$ . Choosing  $M=L-K$  as in FIG. 9 leads to an architecture where the critical path is reduced to be of the same order as in MLSE (c.f. FIG. 6) and does not depend on the channel memory  $L$ . In a further variation, the precomputed partial ISI estimates in the pipelined DFU 910 may be processed in groups of taps, with a precomputation for all possible data symbol combinations corresponding to the groups of taps and selection for each group, as would be apparent to a person of ordinary skill in the art.

FIGS. 11 and 12 illustrate alternate implementations of the RSSEs shown in FIGS. 8 and 9, respectively, where pipeline registers are placed differently (now before Mux at stage 1125 and 1225, respectively) using a cut-set or re-timing transformation technique. For a

more detailed discussion of cut-set or re-timing transformation techniques, see P. Pirsch, Architectures for Digital Signal Processing, New York, Wiley (1998), incorporated by reference herein. The present invention encompasses all derivations that can be achieved using such transformations, as would be apparent to a person of ordinary skill in the art.

## 5 JOINT POSTCURSOR EQUALIZATION AND TRELLIS DECODING FOR 1000BASE-T GIGABIT ETHERNET

An exemplary embodiment employs the 1000BASE-T physical layer standard that specifies Gigabit Ethernet over four pairs of Category 5 unshielded twisted pair (UTP) copper cabling, as described in M. Hatamian et al., "Design Considerations for Gigabit Ethernet  
10 1000Base-T Twisted Pair Transceivers," Proc. IEEE Custom Integrated Circuits Conf. (CICC), Santa Clara, CA, 335-342 (May 1998); or K. Azadet, "Gigabit Ethernet Over Unshielded Twisted Pair Cables," Proc. Int. Symp. VLSI Technology, Systems, Applications (VLSI-TSA), Taipei, 167-170 (June 1999). It is noted that hereinafter, all variables will be defined in a new way. Although the meaning of variables used in this second part of this detailed description may  
15 be related to the definition of the variables in the previous part of the description, they might not exactly have the same meaning. All variables used hereinafter, however, will be described and defined in a precise way and their meaning is valid for this second part of the detailed description only.

The throughput of 1Gb/s is achieved in 1000BASE-T by full duplex transmission  
20 of pulse amplitude modulated signals with the five levels  $\{-2, -1, 0, 1, 2\}$  (PAM5) resulting in a data rate of 250Mb/s per wire pair. By grouping four PAM5 symbols transmitted over the four different wire channels, a four-dimensional (4D) symbol is formed which carries eight information bits.

Thus, the symbol rate is 125Mbaud/s, which corresponds to a symbol period of  
25 8ns. To achieve a target bit error rate of at less than  $10^{-10}$ , the digital signal processor (DSP) section of a 1000BASE-T receiver must cancel intersymbol interference (ISI), echo and near-end crosstalk (NEXT). 1000BASE-T improves the noise margin by employing trellis-coded modulation (TCM). For a detailed discussion of trellis-coded modulation techniques, see, for  
30 example, G. Ungerboeck, "Trellis-Coded Modulation With Redundant Signal Sets, Parts I and II," IEEE Commun. Mag., Vol. 25, 5-21 (Feb. 1987), incorporated by reference herein.

For coding purposes, the 1D PAM5 symbols are partitioned into two one dimensional (1D) subsets  $A=\{-1,1\}$  and  $B=\{-2,0,2\}$ . By grouping different combinations of the 1D subsets together which are transmitted over the four wire pairs, the eight 4D subsets  $S_0, S_1, \dots, S_8$  are formed. The 8-state, radix-4 code trellis specified by 1000BASE-T is shown in FIG. 13.  $\rho_n$  in Fig. 13 denotes the state of the trellis code at time  $n$  (i.e.,  $\rho_n$  is no longer defined by equation (2), as noted at the beginning of this section). Each transition in the trellis diagram 1300 corresponds to one of the specified eight 4D subsets. There are 64 parallel transitions per state transition. Due to the 4D subset partitioning and labeling of the transitions in the code trellis, the minimum Euclidean distance between allowed sequences is  $\Delta^2=4$  which corresponds to an asymptotic coding gain of 6 dB ( $10\log 4$ ) over uncoded PAM5 in an ISI free channel.

In a 1000BASE-T receiver, feedforward equalizers, echo and NEXT cancellers remove precursor ISI, echo and NEXT respectively. The remaining DSP processing removes the postcursor ISI, which typically spans 14 symbol periods, and decodes the trellis code. It has been shown in E.F. Haratsch, "High-Speed VLSI Implementation of Reduced Complexity Sequence Estimation Algorithms With Application to Gigabit Ethernet 1000BASE-T," Proc. Int. Symp. VLSI Technology, Systems, Applications (VLSI-TSA), Taipei, 171-174 (June 1999) that parallel decision-feedback decoding, a special case of reduce-state sequence estimation, M.V. Eyuboglu and S.U. Qureshi, "Reduced-State Sequence Estimation for Coded Modulation on Intersymbol Interference Channels," IEEE J. Sel. Areas Commun., Vol. 7, 989-95 (Aug. 1989), offers the best trade-off for this task with respect to SNR performance, hardware complexity and critical path. However, the integration of a 125MHz, 14-tap parallel decision-feedback decoder (PDFD) is quite challenging because of the critical path problem.

A simplified postcursor equalization and trellis decoding structure was presented in E.F. Haratsch and K. Azadet, "A Low Complexity Joint Equalizer and Decoder for 1000BASE-T Gigabit Ethernet," Proc. IEEE Custom Integrated Circuits Conf. (CICC), Orlando, 465-68 (May 2000), where decision-feedback prefilters shorten the postcursor impulse response to one postcursor. Then exhaustive precomputation of all possible 1D branch metrics is possible, substantially reducing the critical path of the remaining 1-tap PDFD. However, the postcursor equalization and trellis decoding structure suffers from a performance degradation of 1.3dB

compared to a 14-tap PDFD.

The present invention thus provides a pipelined 14-tap PDFD architecture, which operates at the required processing speed of 125MHz without any coding gain loss. To achieve this, the look-ahead technique discussed above for uncoded signals impaired by ISI, where the underlying trellis has no parallel state transitions, is extended to trellis codes with parallel transitions like the one specified in 1000BASE-T. The processing blocks of the disclosed architecture, which differ from a conventional PDFD design, are described below.

#### Parallel Decision-Feedback Decoding Algorithm

Parallel decision-feedback decoding combines postcursor equalization with TCM decoding by computing separate ISI estimates for each code state before applying the well known Viterbi algorithm (see, e.g., G. D. Forney, Jr., "The Viterbi Algorithm," Proc. IEEE, Vol. 61, 268-78 (Mar. 1973)) to decode the trellis code. An ISI estimate for wire pair  $j$  and code state  $\rho_n$  at time  $n$  is given by

$$u_{n,j}(\rho_n) = \sum_{i=1}^{14} f_{i,j} \hat{a}_{n-i,j}(\rho_n),$$

where  $\{f_{i,j}\}$  are the postcursor channel coefficients for wire pair  $j$  and  $\hat{a}_{n-i,j}(\rho_n)$  is the  $j$ -th dimension of the 4D survivor symbol  $\hat{\underline{a}}_{n-i}(\rho_n) = (\hat{a}_{n-i,1}(\rho_n), \hat{a}_{n-i,2}(\rho_n), \hat{a}_{n-i,3}(\rho_n), \hat{a}_{n-i,4}(\rho_n))$ , which belongs to the survivor sequence into  $\rho_n$  and corresponds to time  $n-i$ . As there are eight code states and four wire pairs, 32 ISI estimates are calculated at each decoding step. In a straight-forward PDFD implementation, the calculation of the ISI estimates in the decision-feedback unit introduces a recursive loop, which also includes the branch metric unit (BMU), add-compare-select unit (ACSU) and survivor memory unit (SMU). As the clock rate is 125MHz in 1000BASE-T, there are only 8ns available for the operations along this critical path. As conventional pipelining techniques cannot be applied to improve throughput due to the recursive nonlinear structure of this loop, it is extremely challenging to implement a 125MHz, 14-tap PDFD for 1000BASE-T Gigabit Ethernet. When a state-parallel 14-tap PDFD is implemented using VHDL and synthesis in 3.3V 0.16μm standard cell CMOS process, the design would only achieve a throughput of approximately 500Mb/s, and the hardware complexity would be 158kGates. In the following, a pipelined 14-tap PDFD architecture is disclosed that achieves the required throughput of 1Gb/s without sacrificing coding gain performance.



### Pipelined 14-Tap PDFD Architecture

The parallel decision-feedback decoding algorithm was reformulated above such that pipelining of the computation of the ISI estimates and branch metrics is possible. ISI estimates and branch metrics are precomputed in a look-ahead fashion to bring the DFU and BMU out of the critical loop (see FIGS. 8 and 9 and corresponding discussion). Using ACS decisions to prune the look-ahead computation tree mitigates the exponential growth of the computational complexity with respect to the look-ahead depth. The above discussion only addressed the case where parallel decision-feedback decoding or other RSSE variants are used for equalization (and trellis decoding) of signals impaired by ISI, where the underlying trellis has no parallel state transitions.

In the following discussion, the look-ahead computation concept discussed above is extended to systems where the parallel decision-feedback decoding algorithm or other RSSE variants are used for equalization and/or trellis decoding where the underlying trellis has parallel state transitions. In particular, an exemplary pipelined, 14-tap PDFD architecture with look-ahead depth two is presented which meets the throughput requirement of 1000BASE-T. The present invention can be generalized to other look-ahead depths, trellis codes, modulation schemes, RSSE variants and number of postcursor taps, as would be apparent to a person of ordinary skill in the art.

The disclosed pipelined PDFD architecture, which decodes the 1000BASE-T trellis code and equalizes the ISI due to 14 postcursors is shown in FIG. 14. Speculative ISI estimates which are used for the ACS decisions corresponding to state transitions  $\{\rho_{n+2}\} \rightarrow \{\rho_{n+3}\}$  are computed in the look-ahead DFU (LA-DFU) 1412 using information already available at time  $n$ , i.e., two clock cycles ahead of time. Therefore, the look-ahead depth is two. The appropriate ISI estimates are selected in the ISI-multiplexer unit (ISI-MUXU) 1416 based on ACS decisions (from 1440) and survivor symbols (from 1450). Speculative 1D branch metrics are precomputed one decoding step in advance in the 1D-LA-BMU 1424. Again, ACS decisions and survivor symbols are used to select the appropriate 1D branch metrics in the 1D-BM-MUXU 1428. The selected 1D branch metrics are added up in the 4D-BMU 1430 to compute the 4D branch metrics, which correspond to state transitions of the code trellis 1300 shown in FIG. 13. The best survivor path for each code state is determined in the ACSU 1440, and the eight survivor paths

are stored in the SMU 1450.

Compared to a conventional PDFD implementation as described in E. F. Haratsch, "High-Speed VLSI Implementation of Reduced Complexity Sequence Estimation Algorithms With Application to Gigabit Ethernet 1000 Base-T," Int'l Symposium on VLSI Technology, Systems, and Applications, Taipei (Jun. 1999), the DFU and 1D-BMU are outside the critical loop, as there is a pipeline stage 1418 between the DFU and 1D-BMU and another pipeline stage 1429 between the 1D-BMU and 4D-BMU. The critical path in the architecture of FIG. 14 includes only the 4D-BMU 1430, ACSU 1440 and SMU 1450. The contribution of the 1D-BM-MUXU 1428 and ISI-MUXU 1416 to the critical path is low. Therefore, the proposed PDFD architecture achieves a throughput twice as high as a conventional PDFD implementation. The proposed PDFD architecture differs from the pipelined structure developed for trellises without parallel state transitions in FIGS. 8 and 9 with respect to the selection of the appropriate ISI estimates and 1D branch metrics in the ISI-MUXU 1416 and 1D-BM-MUXU 1428. As 1000BASE-T employs TCM with parallel state transitions, not only ACS decisions, but also the most recent survivor symbols are required for the selection of the appropriate values as there is not a unique relationship between ACS decisions and survivor symbols. In the following, the implementation of the DFU 1410, 1D-BMU 1420 and SMU 1450 are described in detail. The implementation of the 4D-BMU 1430 and ACSU 1440 is the same as in a conventional PDFD and is already described in E.F. Haratsch and K. Azadet, "A Low Complexity Joint Equalizer and Decoder for 1000BASE-T Gigabit Ethernet," Proc. IEEE Custom Integrated Circuits Conf. (CICC), Orlando, 465-468 (May 2000).

#### Decision-Feedback Unit

Exhaustive precomputation of ISI estimates is not feasible in 1000BASE-T without prefiltering as the number of possible ISI estimates grows exponentially with the number of postcursors. As there are 14 postcursors, four wire pairs and PAM5 modulation is being used, there are in total  $4 \times 5^{14} \approx 2 \times 10^{10}$  possible ISI estimates, which must be precomputed. Precomputing ISI estimates using a limited look-ahead depth reduces the complexity. The exponential growth of the number of precomputed ISI estimates is mitigated as the precomputation is not completely decoupled from the ACS and survivor symbol decisions. Survivor symbols available at time  $n$  are used for the computation of ISI estimates corresponding

to state transitions  $\{\rho_{n+2}\} \rightarrow \{\rho_{n+3}\}$ . Then, the look-ahead computation tree is pruned using ACS and survivor symbol decisions available at time  $n$ .

#### Look-Ahead Computation of ISI Estimates (LA-DFU)

An estimate  $v_{n+2,j}(\rho_n)$  for the partial ISI due to the channel coefficients  $\{f_{3,j}, f_{4,j}, \dots, f_{14,j}\}$  which corresponds to a state transition  $\rho_{n+2} \rightarrow \rho_{n+3}$  can be calculated by using the symbols from the survivor path into state  $\rho_n$  which are available at time  $n$ :

$$v_{n+2,j}(\rho_n) = -\sum_{i=1}^{12} f_{i+2,j} \hat{a}_{n-i,j}(\rho_n).$$

A speculative partial ISI estimate  $\tilde{u}_{n+2,j}(\rho_n, \tilde{a}_{n,j})$ , which also considers the ISI due to  $f_{2,j}$  and assumes that  $\tilde{a}_{n,j}$  is the 1D symbol for the corresponding transition  $\rho_n \rightarrow \rho_{n+1}$  is calculated as

$$\tilde{u}_{n+2,j}(\rho_n, \tilde{a}_{n,j}) = v_{n+2,j}(\rho_n) - f_{2,j} \tilde{a}_{n,j}.$$

As there are five possibilities for  $\tilde{a}_{n,j}$  due to the PAM5 modulation, five different partial ISI estimates must be computed per code state and wire pair in the LA-DFU 1412 as shown in FIG. 15. In total, 160 (8x4x5) such ISI estimates are precomputed in the LA-DFU 1412.

#### Selection of ISI Estimates (ISI-MUXU)

The appropriate partial ISI estimate  $v_{n+2,j}(\rho_{n+1})$  which considers the symbols from the survivor path into  $\rho_{n+1}$  and the channel coefficients  $\{f_{2,j}, f_{3,j}, \dots, f_{14,j}\}$  can be selected among the precomputed partial ISI estimates  $\tilde{u}_{n+2,j}(\rho_n, \tilde{a}_{n,j})$  when the best survivor path into state  $\rho_{n+1}$  and the corresponding 4D survivor symbol  $\hat{a}_{n,j}(\rho_{n+1})$  become available. This selection in the ISI-MUXU 1416 is shown in FIG. 16 for a particular wire pair  $j$  and state  $\rho_{n+1} = 0$ . The partial ISI estimate  $v_{n+2,j}(\rho_{n+1} = 0)$  is selected among 20 (4x5) precomputed partial ISI estimates  $\{\tilde{u}_{n+2,j}(\rho_n, \tilde{a}_{n,j})\}$ ,  $\{\rho_n\} \rightarrow \rho_{n+1} = 0$ , as there are the four contender paths from the states  $\rho_n = 0, 2, 4$  and 6 leading into state  $\rho_{n+1} = 0$ . Also, for each of these contender paths leading into state  $\rho_{n+1}$ , five different partial ISI estimates  $\tilde{u}_{n+2,j}(\rho_n, \tilde{a}_{n,j})$  corresponding to different values for  $\tilde{a}_{n,j}$  are possible. As shown in FIG. 16, the selection of the appropriate partial ISI estimate  $v_{n+2,j}(\rho_{n+1})$  is performed in two stages. First, the ACS decision  $d_n(\rho_{n+1})$  selects the five speculative partial ISI

estimates, which correspond to the selected survivor path into  $\rho_{n+1}$ , but assume different values for  $\tilde{a}_{n,j}$ . Then, the survivor symbol  $\hat{a}_{n,j}(\rho_{n+1})$  selects the appropriate partial ISI estimate  $v_{n+2,j}(\rho_{n+1})$ , which assumed  $\hat{a}_{n,j}(\rho_{n+1})$  as value for  $\tilde{a}_{n,j}$ . Both  $d_n(\rho_{n+1})$  and  $\hat{a}_{n,j}(\rho_{n+1})$  become available at the end of the clock cycle corresponding to state transitions  $\{\rho_n\} \rightarrow \{\rho_{n+1}\}$ . The output of the ISI-MUXU is 32 (8x4) partial ISI estimates  $v_{n+2,j}(\rho_{n+1})$ , as there are eight states and four wire pairs.

#### 1D Branch Metric Unit

The 1D-BMU 1420 consists of two processing blocks. The 1D-LA-BMU 1424 takes the partial ISI estimates  $\{v_{n+1,j}(\rho_n)\}$  computed in the DFU to calculate speculative 1D branch metrics. In the 1D-BM-MUXU 1428, the appropriate 1D branch metrics are selected using ACS decisions and corresponding survivor symbols.

#### Look-ahead computation of 1D branch metrics (1D-LA-BMU)

The 1D-LA-BMU 1424 precomputes speculative 1D branch metrics which are then needed in the 4D-BMU 1430 one clock cycle later. Input into the 1D-LA-BMU 1424 are the partial ISI estimates  $\{v_{n+1,j}(\rho_n)\}$ , which correspond to trellis transitions  $\{\rho_{n+1}\} \rightarrow \{\rho_{n+2}\}$ . These ISI estimates consider the channel coefficients  $\{f_{2,j}, f_{3,j}, \dots, f_{14,j}\}$  and the symbols from the survivor path into state  $\rho_n$ . A speculative partial ISI estimate  $\tilde{u}_{n+1,j}(\rho_n, \tilde{a}_{n,j})$ , which also considers the ISI due to the channel coefficient  $f_{1,j}$  and assumes that  $\tilde{a}_{n,j}$  is the 1D symbol corresponding to the transition  $\rho_n \rightarrow \rho_{n+1}$  is given by:

$$\tilde{u}_{n+1,j}(\rho_n, \tilde{a}_{n,j}) = v_{n+1,j}(\rho_n) - f_{1,j} \tilde{a}_{n,j}.$$

The speculative 1D branch metric for a transition from state  $\rho_{n+1}$  under the symbol  $a_{n+1,j}$  assuming that the corresponding survivor path contains the survivor sequence into state  $\rho_n$  and is extended by the symbol  $\tilde{a}_{n,j}$  to reach state  $\rho_{n+1}$  is given by

$$\begin{aligned} \tilde{\lambda}_{n+1,j}(z_{n+1,j}, a_{n+1,j}, \rho_n, \tilde{a}_{n,j}) = \\ (z_{n+1,j} - a_{n+1,j} + \tilde{u}_{n+1,j}(\rho_n, \tilde{a}_{n,j}))^2. \end{aligned}$$

The precomputation of speculative 1D branch metrics for a particular initial state  $\rho_n$  and wire pair  $j$  is shown in FIG. 17, where the slicers calculate the difference between the slicer input and the closest symbol in the 1D subsets A and B, respectively. As there are four

wire pairs, eight code states, five possibilities for  $\tilde{a}_{n,j}$  (due to the PAM5 modulation), and two possibilities for  $a_{n+1,j}$  (A-type or B-type 1D symbol), in total 320 (8x4x5x2) different speculative 1D branch metrics are precomputed in the 1D-LA-BMU 1424.

#### Selection of 1D branch metrics (1D-BM-MUXU)

The appropriate 1D branch metric  $\lambda_{n+1,j}(z_{n+1,j}, a_{n+1,j}, \rho_{n+1})$  which corresponds to a transition from state  $\rho_{n+1}$  under the 1D symbol  $a_{n+1,j}$  is selected among 4x5=20 precomputed 1D branch metrics  $\tilde{\lambda}_{n+1,j}(z_{n+1,j}, a_{n+1,j}, \rho_n, \tilde{a}_{n,j})$  as there are four path extensions from different states  $\{\rho_n\}$  into  $\rho_{n+1}$  and five possibilities for  $\tilde{a}_{n,j}$  due to the PAM5 modulation. The selection of a particular  $\lambda_{n+1,j}(z_{n+1,j}, a_{n+1,j}, \rho_{n+1})$  in the 1D-BM-MUXU 1428 is performed using the same multiplexer structure as shown in FIG. 16. First, the ACS decision  $d_n(\rho_{n+1})$  determines the five speculative 1D branch metrics, which correspond to the state  $\rho_n$  being part of the survivor path into  $\rho_{n+1}$ . Then, the survivor symbol  $\hat{a}_{n,j}(\rho_{n+1})$  selects among these five metrics the one which assumed  $\hat{a}_{n,j}(\rho_{n+1})$  as value for  $\tilde{a}_{n,j}$ . The 1D-BM-MUXU 1428 selects in total 64 (8x4x2) actual 1D branch metrics, as there are eight states, four wire pairs and the two 1D subset types A and B.

#### Survivor Memory Unit

The merge depth of the exemplary 1000BASE-T trellis code is 14. The SMU must be implemented using the register-exchange architecture described in R. Cypher and C.B. Shung, "Generalized Trace-Back Techniques for Survivor Memory Management in the Viterbi Algorithm," J. VLSI Signal Processing, Vol. 5, 85-94 (1993), as the survivor symbols corresponding to the time steps  $n-12, n-11, \dots, n$  are needed in the DFU without delay and the latency budget specified in the 1000BASE-T standard is very tight. The proposed register-exchange architecture with merge depth 14 is shown in FIG. 18, where only the first row storing the survivor sequence corresponding to state zero is shown.  $\underline{S}X_n(\rho_n)$  denotes the 4D symbol decision corresponding to 4D subset SX and a transition from state  $\rho_n$ . The multiplexers in the first column select the 4D survivor symbols  $\{\hat{\underline{a}}_n(\rho_{n+1})\}$ , which are part of the survivor path into  $\{\rho_{n+1}\}$ . These 4D survivor symbols are required in the ISI-MUXU 1416 and 1D-BM-MUXU 1428 to select the appropriate partial ISI estimates and 1D branch metrics, respectively. The survivor symbols  $\{\hat{\underline{a}}_{n-1}(\rho_n), \hat{\underline{a}}_{n-2}(\rho_n), \dots, \hat{\underline{a}}_{n-12}(\rho_n)\}$  which are stored in the registers corresponding to

the first, second, ... 12th column are used in the LA-DFU 1412 to compute the partial ISI estimates  $v_{n+2,j}(\rho_n)$ .

It is to be understood that the embodiments and variations shown and described herein are merely illustrative of the principles of this invention and that various modifications  
5 may be implemented by those skilled in the art without departing from the scope and spirit of the invention.

FIG. 14 is a schematic diagram of the system.